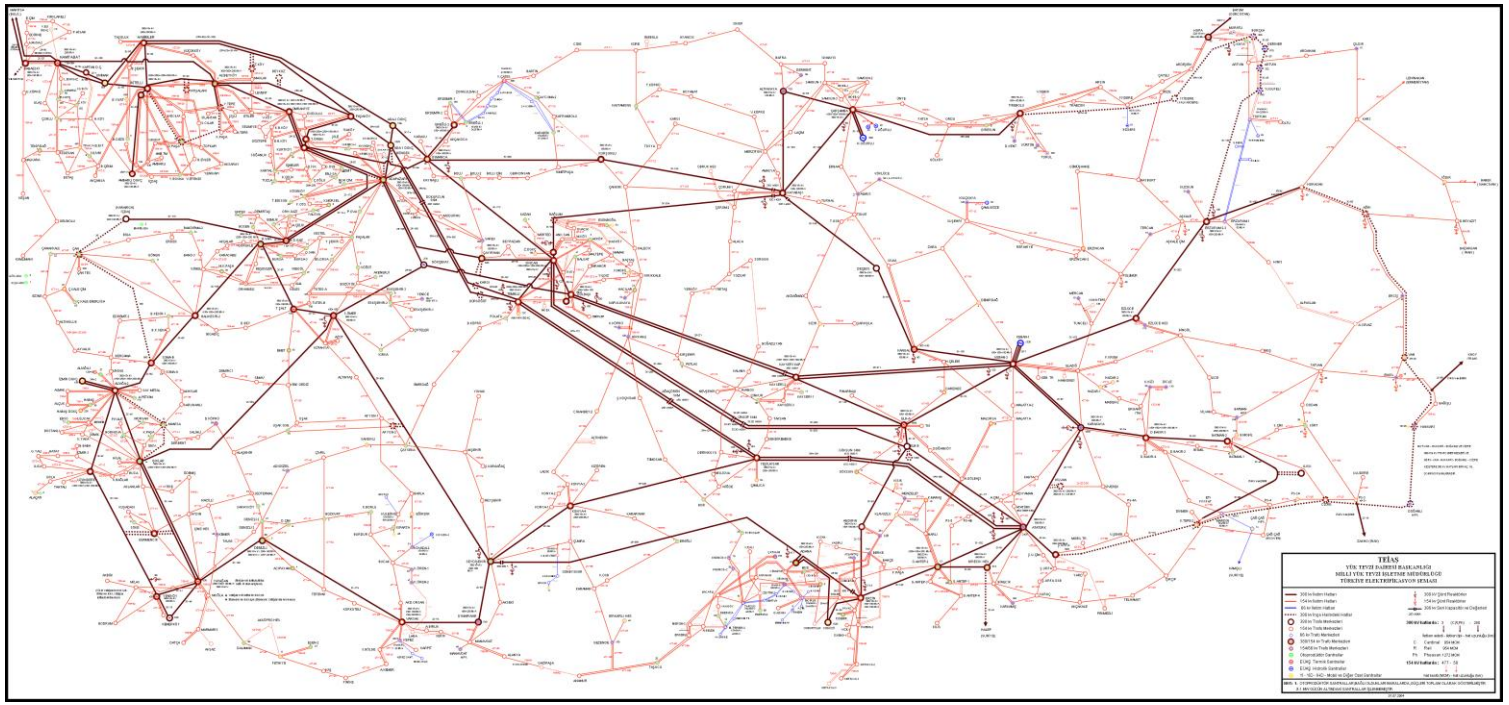
CASK – Open Source Custom Architectures for Sparse Kernels

Paul Grigoras, Pavel Burovskiy, Wayne Luk

Imperial College London

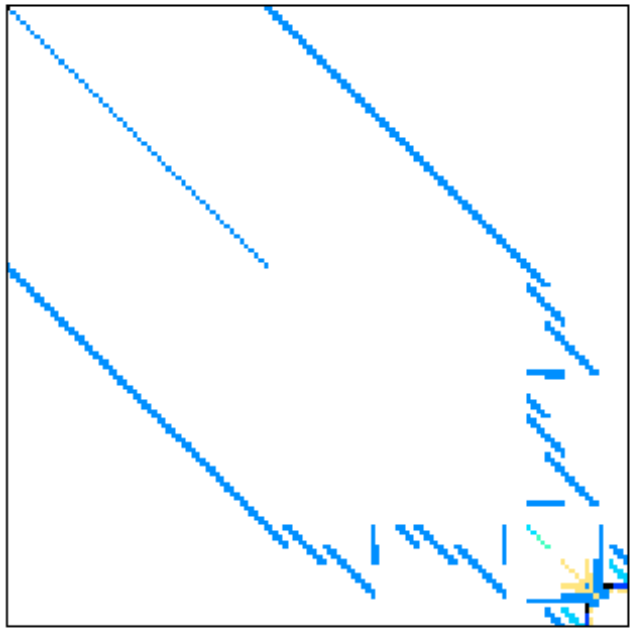
What is Sparse Algebra?

- **Sparse Algebra** = Solving Sparse Linear & Nonlinear Systems of Equations
  - Physical systems are often modelled as large, sparse sets of nonlinear equations
  - **Sparsity** can be exploited to improve performance & reduce storage size



Turkish Power Distribution Grid

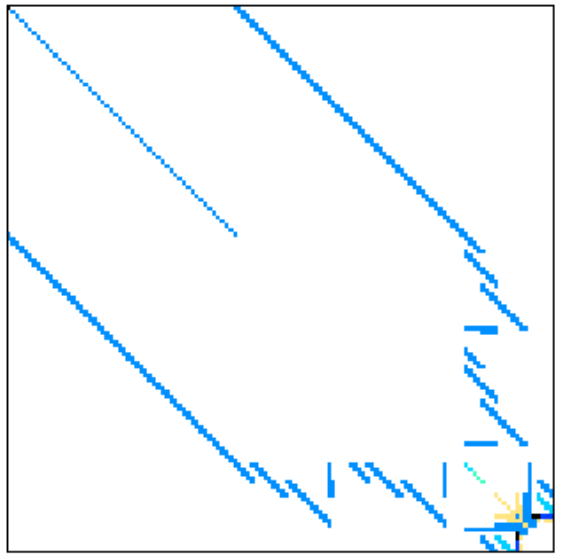
Source: www.geni.org



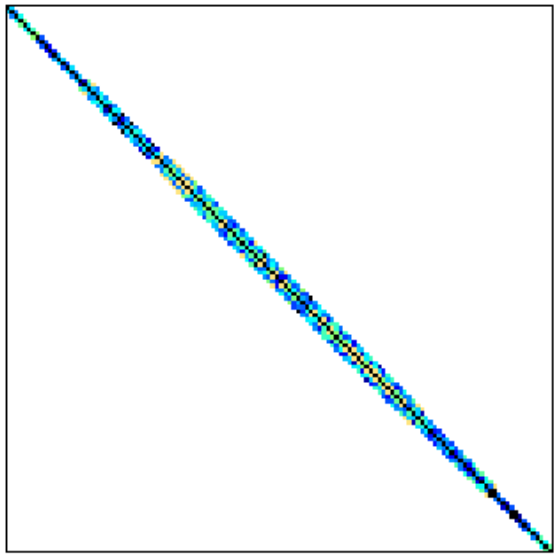
Corresponding Sparse Representation

Source: University of Florida Sparse Matrix Collection

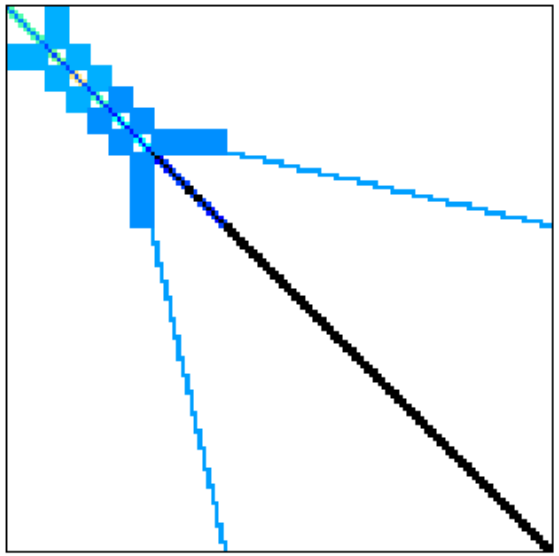
Need for Customisation in Sparse Algebra



Power System Simulation



Computational Fluid Dynamics



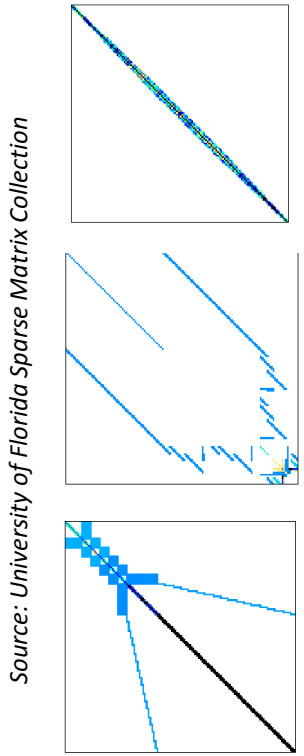
Circuit Simulation Problem

Source: University of Florida Sparse Matrix Collection

For any sparse system – sparsity pattern, value range, order and nature of the algorithm may be used to improve performance

➔ **Custom Computing for Sparse Algebra**

CASK – Custom Architectures for Sparse Kernels



Source: University of Florida Sparse Matrix Collection

Sparse Matrix Benchmark

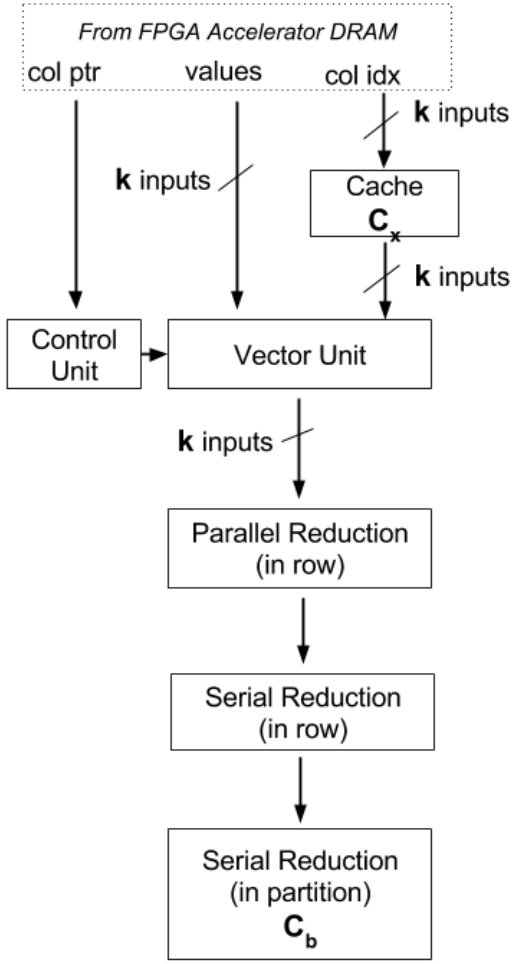
CASK

Custom Architectures for Sparse Kernels

Open Source Tool

http://caskorg.github.io/cask/

FPL14, FCCM15, FPGA16



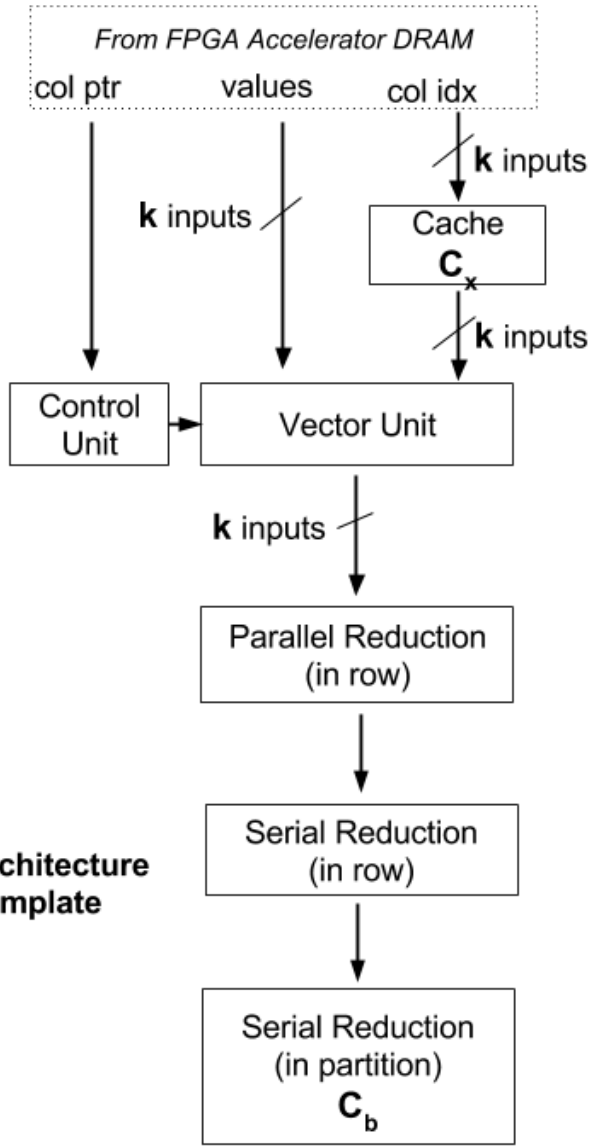
Generic Architecture

Source: www.nallatech.com

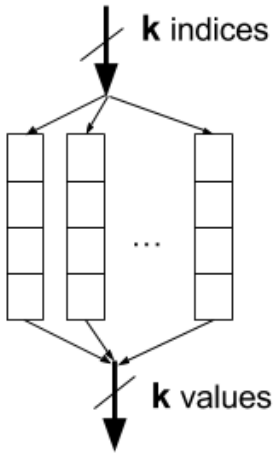
FPGA Implementation

Custom Architectures for Sparse Kernels

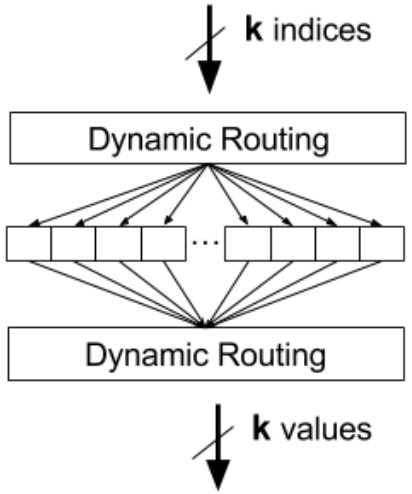
Architecture Template



Cache Structure (*Replicated vs Dynamic*) - trade-off storage size/maximum order vs retrieval rate

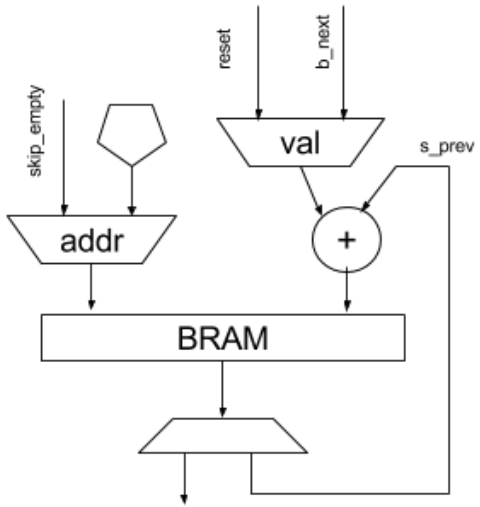


k-way replication
+ k reads / cycle
- k x V_b storage
- increases traffic
- increases resource usage

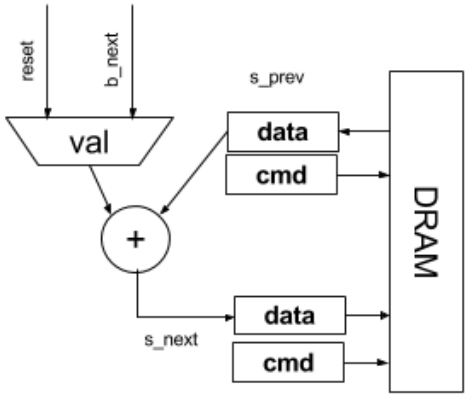


p-way replication (p << k)
- conflicts ⇒ <= k reads / cycle
+ p x V_b storage
+ larger V_b ⇒ reduced traffic
+ reduced resource usage

Result Cache (*On-Chip vs Off-Chip*) - trade-off latency/performance vs problem size/resource usage



+ ~13 cycles latency
+ support sparse vector
- n < 100K
- more BRAMs



- ~30K cycles latency
+ n > 100K
+ fewer BRAMs

Impact of Automated Customisation in Sparse Algebra

Table 1: Required architectures for each matrix, produced in our approach (for Maxeler Vectis)

Matrix				Architecture				Place & Route	Peak Performance	
Name	Order	Nonzeros	Nnz/row	Cx	k	N_p	Cb	Logic/DSP/BRAM %	GB/s	GFLOPs
dense	2048	4194304	2048.00	2048	16	2	2048	42.63 / 23.41 / 43.14	38.4	6.30
psmigr_2	3140	540022	171.98	4096	16	2	3584	42.02 / 23.41 / 54.23	38.4	4.76
raefsky1	3242	294276	90.77	4096	16	2	3584	42.02 / 23.41 / 54.23	38.4	3.99
rma10	46835	2374001	50.69	7168	16	2	47104	42.91 / 23.41 / 84.87	38.4	1.63
consph	83334	3046907	36.56	9216	8	2	83456	37.46 / 12.30 / 82.61	19.2	1.37
cant	62451	2034917	32.58	11264	8	2	62464	37.15 / 12.30 / 80.92	19.2	1.60
shipsec1	140874	3977139	28.23	14336	16	1	141312	30.24 / 11.71 / 79.65	19.2	0.78
torso2	115967	1033473	8.91	15360	16	1	116224	30.97 / 11.71 / 78.62	19.2	0.19
t2d_q9_A_01	9801	87025	8.88	10240	8	2	10240	36.24 / 12.30 / 60.81	19.2	0.87
epb1	14734	95053	6.45	15360	8	2	14848	37.06 / 12.30 / 75.94	19.2	0.69
mac_econ	206500	1273389	6.17	15360	8	1	206848	27.31 / 6.10 / 73.07	9.6	0.08
scircuit	170998	958936	5.61	14336	16	1	171008	30.39 / 11.71 / 84.54	19.2	0.08
dw8192	8192	41746	5.10	8192	8	3	8192	45.59 / 18.45 / 78.57	28.8	0.68

Fully Support Wide Range Of Problems

Generate Optimised Architectures

Improve Performance and Resource Utilisation

Future: application- and system-specific optimisations; run-time reconfiguration; energy reduction

Consider Instance Specific Custom Computing for Sparse Matrix Problems!

http://caskorg.github.io/cask